package com.twitter.tweetypie

package store

import com.twitter.finagle.stats.RollupStatsReceiver

import com.twitter.servo.util.MemoizingStatsReceiver

/\*\*

\* Records some stats about inserted tweets. Tweets are currently classified by three criteria:

\*

\* - tweet type: "tweet" or "retweet"

\* - user type: "stresstest", "protected", "restricted", or "public"

\* - fanout type: "nullcast", "narrowcast", or "usertimeline"

\*

\* A counter is incremented for a tweet using those three criteria in order. Counters are

\* created with a RollupStatsReceiver, so counts are aggregated at each level. Some

\* example counters are:

\*

\* ./insert

\* ./insert/tweet

\* ./insert/tweet/public

\* ./insert/tweet/protected/usertimeline

\* ./insert/retweet/stresstest

\* ./insert/retweet/public/nullcast

\*/

trait TweetStatsStore extends TweetStoreBase[TweetStatsStore] with InsertTweet.Store {

def wrap(w: TweetStore.Wrap): TweetStatsStore =

new TweetStoreWrapper(w, this) with TweetStatsStore with InsertTweet.StoreWrapper

}

object TweetStatsStore {

def apply(stats: StatsReceiver): TweetStatsStore = {

val rollup = new MemoizingStatsReceiver(new RollupStatsReceiver(stats))

val inserts = rollup.scope("insert")

def tweetType(tweet: Tweet) =

if (getShare(tweet).isDefined) "retweet" else "tweet"

def userType(user: User) =

if (user.roles.exists(\_.roles.contains("stresstest"))) "stresstest"

else if (user.safety.exists(\_.isProtected)) "protected"

else if (user.safety.exists(\_.suspended)) "restricted"

else "public"

def fanoutType(tweet: Tweet) =

if (TweetLenses.nullcast(tweet)) "nullcast"

else if (TweetLenses.narrowcast(tweet).isDefined) "narrowcast"

else "usertimeline"

new TweetStatsStore {

override val insertTweet: FutureEffect[InsertTweet.Event] =

FutureEffect[InsertTweet.Event] { event =>

inserts

.counter(

tweetType(event.tweet),

userType(event.user),

fanoutType(event.tweet)

)

.incr()

Future.Unit

}

}

}

}